```
;*********************************************************************
;*                                                                   *
;*              M I C R O C O M P U T E R    A  5 1 2 0             *
;*                                                                   *
;*   BIOS      Vers.1.2   01.06.85       (SCPX 1526)                 *
;*                                                                   *
;*   Zeichn.-Nr. 083-3-XXX-XXX/XX                                    *
;*               1.62.XXXXXX.X/XX                                    *
;*                                                             *     *
;*                                                                   *
;*********************************************************************
;
title BIOS    Vers.1.2  01.06.85; Var.: BIOS722 (K7637,BAB2,V 1.2)
;
;
;
msize equ   57    ;scp version memory size in kilobytes
;
bias  equ   (msize-20)*1024
ccp   equ   3400h+bias  ;base of ccp
bdos  equ   ccp+806h    ;base of bdos
bios  equ   ccp+1600h   ;base of bios
cdisk equ   0004h       ;current disk number 0=A,...,15=P
iobyte      equ   0003h       ;intel i/o byte
hstsiz      equ   1024        ;host block size
bufmsk      equ   (hstsiz/128)-1    ;buffer mask
vdu   equ   0f800h
intab equ   0f740h            ;beginn INT. table
;
      aseg
      org   100h        ;origin of this program
      .phase      bios
      .z80
;
      page  68
;     jump vector for individual subroutines
      jp    boot        ;cold start
wboote:     jp    wboot       ;warm start
      jp    const       ;console status
      jp    conin       ;console character in
      jp    conout            ;console character out
      jp    list        ;list character out
      jp    punch       ;punch character out
      jp          reader            ;reader character out
      jp    home        ;mOve head to home position
      jp    seldsk            ;select disk
      jp    settrk            ;set track number
      jp    setsec            ;set sector number
      jp    setdma            ;set dma-adress
      jp    read        ;read disk
      jp    write       ;write disk
      jp    listst            ;return list status
      jp    sectran           ;sectortranslate
;
sign: db    'BW'        ;BWK signon
vers: db    ' 12'       ;vers.nr. = 1.2
cpos: dw    vdu+80            ;current vdu pointer
errad:      dw    erradr            ;adr. error routine BContr.
idadr:      dw    idm         ;adr. IDM check field
wcdb: dw    cdbw1       ;adr. wboot cdb
inter:      dw    intab       ;INT. table for DISKIO
dskio:      dw    diskio            ;adr. diskio
dskb: dw    dpbase            ;adr 1. dsk param. haeder
```

```
errc: db    8           ;number of retries if r/w errors
treiwba:dw  treiwb            ;driver-wboot-address
reserv:     db    0
      dw    0
      dw    0
      dw    0
tranad:     dw    trand       ;adr. trand block
cdb:        ds    13          ;CDB for DISKIO
bufad:      dw    hstbuf            ;adr hostbuffer
dmaad:      ds    2           ;here save DMA
var:  db    40h         ;h-nib.:nr. 8" drives/l-nib.:nr. 5" drives
hd:   db    0           ;nr. of Hard Disk drives
dsknum:     db    5           ;number of log. disks in this system

;     cbd's & dmabl's for wboot
cdbw1:      db    6,0,0,0,5,0,26,4,22     ;cdb TR0
      db    39,84h,4,21
dma1: dw    ccp               ;dma TR0
;
cdbw2:      db    46h,0,1,0,1,3,8,15,2    ;cdb TR 1...
      db    39,80h,4,21
dma2: dw    ccp+(22*128)            ;dma TR 1...
;
;     disk parameter haeder for disk 00
dpbase:     dw    trand,0000h
      dw    0000h,0000h
      dw    dirbf,dpb00
      dw    chk00,all00
;     disk parameter haeder for disk 01
      dw    trand,0000h
      dw    0000h,0000h
      dw    dirbf,dpb01
      dw    chk01,all01
;     disk parameter haeder for disk 02
      dw    trand,0000h
      dw    0000h,0000h
      dw    dirbf,dpb02
      dw    chk02,all02
;     disk parameter haeder for disk 03
      dw    trand,0000h
      dw    0000h,0000h
      dw    dirbf,dpb03
      dw    chk03,all03
;     disk parameter haeder for disk 04
      dw    trans,0000h
      dw    0000h,0000h
      dw    dirbf,dpb04
      dw    chk04,all04

dpb00:      ;disk parameter block for disk 00
      dw    64          ;SPT        SCP-sec./track
      db    4           ;BSH  Block shift mask
      db    15          ;BLM  Block mask
      db    0           ;EXM  Extent mask
      dw    299         ;DSM  Number of blocks per disk
      dw    127         ;DRM  Number of DIR entries
      db    11000000b   ;AL0  1. Byte ALL. Vec.
      db    00000000b   ;AL1  2. Byte ALL. Vec.
      dw    32          ;CKS  Number of DIR check bytes
      dw    2           ;OFF  Number of Systemtracks
;
      db    3           ;SPH  Host shift mask
      db    26+8        ;COR  Number of Syst.sec. to corrigate
      db    1           ;HPB  Number of phy. sec. per Hostbuf.
```

```
      db    1           ;DEB  Deblocking marker
      dw    bufad       ;DMA  Adr. Bytehandling (buff or DMA)
      db    00001110b   ;DRD  Drive descriptor
      db    46h         ;COM  Command byte READ
      db    0           ;LUN  Log. Drive & Haed adr.
      db    3           ;N    Phys. Sectorsize
      db    8           ;EOT  Nr. End sec. on Track
      db    15          ;ALB  Number of gap byte
      db    39          ;PRAE Tracknumber Praekompensation on
      db    80H         ;ALC  Adapter & Drive control byte
      db    04          ;SRT  Step rate time
      db    21          ;HLT  Haed load time
      db    08          ;PSPC Number of phys. sec./cylinder
      db    76          ;LCN  Last Cylinder Number

;
dpb01:      ;disk parameter block for disk 01
      dw    64          ;SPT        SCP-sec./track
      db    4           ;BSH  Block shift mask
      db    15          ;BLM  Block mask
      db    0           ;EXM  Extent mask
      dw    299         ;DSM  Number of blocks per disk
      dw    127         ;DRM  Number of DIR entries
      db    11000000b   ;AL0  1. Byte ALL. Vec.
      db    00000000b   ;AL1  2. Byte ALL. Vec.
      dw    32          ;CKS  Number of DIR check bytes
      dw    2           ;OFF  Number of Systemtracks
;
      db    3           ;SPH  Host shift mask
      db    26+8        ;COR  Number of Syst.sec. to corrigate
      db    1           ;HPB  Number of phy. sec. per Hostbuf.
      db    1           ;DEB  Deblocking marker
      dw    bufad       ;DMA  Adr. Bytehandling (buff or DMA)
      db    00001110b   ;DRD  Drive descriptor
      db    46h         ;COM  Command byte READ
      db    1           ;LUN  Log. Drive & Haed adr.
      db    3           ;N    Phys. Sectorsize
      db    8           ;EOT  Nr. End sec. on Track
      db    15          ;ALB  Number of gap byte
      db    39          ;PRAE Tracknumber Praekompensation on
      db    80H         ;ALC  Adapter & Drive control byte
      db    04          ;SRT  Step rate time
      db    21          ;HLT  Haed load time
      db    08          ;PSPC Number of phys. sec./cylinder
      db    76          ;LCN  Last Cylinder Number
;
dpb02:      ;disk parameter block for disk 02
      dw    64          ;SPT        SCP-sec./track
      db    4           ;BSH  Block shift mask
      db    15          ;BLM  Block mask
      db    0           ;EXM  Extent mask
      dw    299         ;DSM  Number of blocks per disk
      dw    127         ;DRM  Number of DIR entries
      db    11000000b   ;AL0  1. Byte ALL. Vec.
      db    00000000b   ;AL1  2. Byte ALL. Vec.
      dw    32          ;CKS  Number of DIR check bytes
      dw    2           ;OFF  Number of Systemtracks
;
      db    3           ;SPH  Host shift mask
      db    26+8        ;COR  Number of Syst.sec. to corrigate
      db    1           ;HPB  Number of phy. sec. per Hostbuf.
      db    1           ;DEB  Deblocking marker
      dw    bufad       ;DMA  Adr. Bytehandling (buff or DMA)
      db    00001110b   ;DRD  Drive descriptor
```

```
        db    46h         ;COM  Command byte READ
        db    2           ;LUN  Log. Drive & Haed adr.
        db    3           ;N    Phys. Sectorsize
        db    8           ;EOT  Nr. End sec. on Track
        db    15          ;ALB  Number of gap byte
        db    39          ;PRAE Tracknumber Praekompensation on
        db    80H         ;ALC  Adapter & Drive control byte
        db    04          ;SRT  Step rate time
        db    21          ;HLT  Haed load time
        db    08          ;PSPC Number of phys. sec./cylinder
        db    76          ;LCN  Last Cylinder Number
;
dpb03:  ;disk parameter block for disk 03
        dw    64          ;SPT        SCP-sec./track
        db    4           ;BSH  Block shift mask
        db    15          ;BLM  Block mask
        db    0           ;EXM  Extent mask
        dw    299         ;DSM  Number of blocks per disk
        dw    127         ;DRM  Number of DIR entries
        db    11000000b   ;AL0  1. Byte ALL. Vec.
        db    00000000b   ;AL1  2. Byte ALL. Vec.
        dw    32          ;CKS  Number of DIR check bytes
        dw    2           ;OFF  Number of Systemtracks
;
        db    3           ;SPH  Host shift mask
        db    26+8        ;COR  Number of Syst.sec. to corrigate
        db    1           ;HPB  Number of phy. sec. per Hostbuf.
        db    1           ;DEB  Deblocking marker
        dw    bufad       ;DMA  Adr. Bytehandling (buff or DMA)
        db    00001110b   ;DRD  Drive descriptor
        db    46h         ;COM  Command byte READ
        db    3           ;LUN  Log. Drive & Haed adr.
        db    3           ;N    Phys. Sectorsize
        db    8           ;EOT  Nr. End sec. on Track
        db    15          ;ALB  Number of gap byte
        db    39          ;PRAE Tracknumber Praekompensation on
        db    80H         ;ALC  Adapter & Drive control byte
        db    04          ;SRT  Step rate time
        db    21          ;HLT  Haed load time
        db    08          ;PSPC Number of phys. sec./cylinder
        db    76          ;LCN  Last Cylinder Number
;
dpb04:  ;disk parameter block for disk 04
        dw    26          ;SPT        SCP-sec./track
        db    3           ;BSH  Block shift mask
        db    7           ;BLM  Block mask
        db    0           ;EXM  Extent mask
        dw    242         ;DSM  Number of blocks per disk
        dw    63          ;DRM  Number of DIR entries
        db    11000000b   ;AL0  1. Byte ALL. Vec.
        db    00000000b   ;AL1  2. Byte ALL. Vec.
        dw    16          ;CKS  Number of DIR check bytes
        dw    2           ;OFF  Number of Systemtracks
;
        db    0           ;SPH  Host shift mask
        db    0           ;COR  Number of Syst.sec. to corrigate
        db    1           ;HPB  Number of phy. sec. per Hostbuf.
        db    0           ;DEB  Deblocking marker
        dw    dmaad       ;DMA  Adr. Bytehandling (buff or DMA)
        db    00000000b   ;DRD  Drive descriptor
        db    06h         ;COM  Command byte READ
        db    3           ;LUN  Log. Drive & Haed adr.
        db    0           ;N    Phys. Sectorsize
        db    26          ;EOT  Nr. End sec. on Track
```

```
      db    04          ;ALB  Number of gap byte
      db    255         ;PRAE Tracknumber Praekompensation on
      db    84H         ;ALC  Adapter & Drive control byte
      db    04          ;SRT  Step rate time
      db    21          ;HLT  Haed load time
      db    26          ;PSPC Number of phys. sec./cylinder
      db    76          ;LCN  Last Cylinder Number
;
;
trand:      ;sector translate vector for DD Floppy & Winchester
      dw    0           ;only marker
;
trans:      ;sector translate vector for SD Floppy
      db    1,7,13,19
      db    25,5,11,17
      db    23,3,9,15
      db    21,2,8,14
      db    20,26,6,12
      db    18,24,4,10
      db    16,22
;
;
;     individual subroutines to perform each function
boot: ;simplest case is to just perform parameter initialization
;     this has been done by the load rom (LDROM)
      di
      ld    a,0f7h
      ld    i,a         ;set I-register to F7H

      ld    hl,intab
      ld    de,intab+1  ;init Int. table with 00H
      ld    bc,vdu-intab-1
      xor   a
      ld    m,a
      ldir

      ld    hl,0        ;init. first 2 KByte with 00H
      ld    de,1
      ld    bc,7ffh
      ld    m,a
      ldir
      ld    sp,100h           ;set stack pointer
      ei

      ld    (prmark),a  ;init "markers" by 00H
      ld    hl,vdu+50H  ;init cpos with F850H
      ld    (cpos),hl
;
      call  keyinit           ;keyboard initialization
;
      ld    hl,msg
      call  prints
      jp    gocpm
prints:     ld    a,m         ;print bios signon msg or error msg
      or    a
      ret   z           ;00H is end marker
      ld    c,a
      push  hl
      call  conout
      pop   hl
      inc   hl
      jr    prints
;
msg:  db    14H,'SCPX 1526 - V 1.2  (52K)',13,10,10,0
```

```
;
;
wboot:      ;load ccp & bdos from disk A
      ld    sp,80h
      ld    hl,cdbw1    ;disk A part 1
      ld    de,(dma1)
      call  gowb
      jr    nz,outwb    ;if error
      ld    hl,cdbw2    ;disk A part 2
      ld    de,(dma2)
      call  gowb
      jr    nz,outwb    ;if no error test readed system
      ld    hl,(ccp+1)
      ld    de,ccp+36ch
      sbc   hl,de
      jr    z,gocpm           ;no error
outwb:      ld    hl,msgout
      call  prints            ;error msg
      ld    a,0ffh
halt: out   (18h),a           ;entriegeln
      jr    halt
;
msgout:     db    0dh,14h,"NO SYSTEM",0
;
gowb: push  de
      ld    de,cdb
      push  de
      ld    bc,13
      ldir              ;move cdbw to cdb
      pop   hl
      pop   de
      call  diskio            ;force read
      ld    a,b
      or    a           ;set Z-flag if no Error
      ret
;
;     end of load operation, now set parameters and go to cp/m
;
gocpm:      ld    hl,bdos           ;bdos entry point
      ld    (6),hl            ;address field of jp at 5 to bdos
treiwb: ld      a,0c3h        ;c3 is a jmp instruction
      ld    (0),a       ;for jmp to wboote
      ld    hl,wboote   ;wboote entry point
      ld    (1),hl            ;set address field for jmp at 0
      ld    (3eh),hl    ;set wboote-address
      ld    (5),a       ;for jmp to bdos
;
      ld    bc,80h            ;default dma address is 80h
      call  setdma            ;
      ld    a,0dfh
      ld    (0f7ebh),a  ;set address field for INT-table beginning
      call  keyrst            ;keyboard reset
;
      xor   a
      ld    (hstact),a  ;host buffer inactiv
      ld    (unacnt),a  ;clear unalloc count
;
      ld    a,(cdisk)   ;get current disk number
      ld    c,a         ;send to the ccp
      jp    ccp         ;go to cp/m for further processing
;
      include     KVP722
;
      ;punch character from register c
```

```
punch:      ld    a,c         ;character to register a
      ret               ;nul subroutine
      ;
      ;read character into register a from reader device
reader:     ld    a,1ah       ;end of file
      ret
      ;
      ;move to the track 00 position of current drive
      ;translate this call into a settrk call with parameter 00
home: ld    bc,0        ;select track 0
      call  settrk
      ld    a,(hstwrt)  ;check for pending write
      or    a
      ret   nz
      ld    (hstact),a  ;clear host active flag
      ret
      ;
      ;SELECT DISK
seldsk:     ld    hl,dsknum   ;max. disk number
      ld    a,c         ;selected disk to A
      cp    m
      ld    hl,0        ;HL = 0 for error on disk number
      ret   nc
      ld    (sekdsk),a  ;disk number correctly selected
      ld    l,a         ;disk number to hl
      add   hl,hl       ;x2
      add   hl,hl       ;x4
      add   hl,hl       ;x8
      add   hl,hl       ;x16
      ld    de,dpbase   ;base of parm block
      add   hl,de       ;hl=dpb (CURDSK)
      ret
      ;
      ;set track given by register bc
settrk:     ld    h,b
      ld    l,c
      ld    (sektrk),hl ;track to seek
      ret
      ;
      ;set sector given by register C
setsec:     ld    a,c
      ld    (seksec),a  ;sector to seek
      ret
      ;
sectran:ex  de,hl       ;hl=.transvector
      ld    a,m
      inc   hl
      or    m
      dec   hl
      jr    z,sectdd
      add   hl,bc
      ld    l,m
      ld    h,0
      ret
sectdd:     ld    h,b
      ld    l,c
      ret
      ;
      ;set dma address given by registers b and c
setdma:     ld    l,c         ;low order address
      ld    h,b         ;high order address
      ld    (dmaad),hl  ;save the address
      ret
      ;
```

```
      ;read the selected cp/m sector
      ;
wrall equ   0           ;write to allocated
wrdir equ   1           ;write to directory
wrual equ   2           ;write to unallocated
      ;
read: call  logvec
      bit   0,(ix+18)
      jr    nz,readd    ;if DEBlocking
      ld    (hstvec),a  ;for SD
      ld    (hstvec+1),hl
      ld    a,(sekdsk)
      ld    (hstdsk),a
      jp    readhst
;
readd:      ld    (sekvec),a
      ld    (sekvec+1),hl
      xor   a
      ld    (unacnt),a
      ld    a,1
      ld    (readop),a  ;read operation
      ld    (rsflag),a  ;must read data
      ld    a,wrual
      ld    (wrtype),a  ;treat as unalloc
      jp    rwoper            ;to perform the read
      ;
      ;write the selected cp/m sector
      ;
write:      ld    hl,wrtype
      ld    m,c
      call  logvec
      bit   0,(ix+18)   ;deblocking marked ?
      jr    nz,writed   ;if yes
      ld    (hstvec),a
      ld    (hstvec+1),hl
      ld    a,(sekdsk)
      ld    (hstdsk),a
      jp    writehst
;
writed:     ld    (sekvec),a
      ld    (sekvec+1),hl
      xor   a
      ld    (readop),a  ;not a read operation
      ld    a,(wrtype)  ;write type in (wrtype)
      cp    wrual       ;write unallocated?
      jp    nz,chkuna   ;check for unalloc
      ;
      ;write to unallocted, set parameters
      ld    a,(ix+3)    ;next unallocated recs
      inc   a
      ld    (unacnt),a
      ld    a,(sekdsk)  ;disk to seek
      ld    (unadsk),a  ;unadsk=sekdsk
      ld    a,(sekvec)
      ld    hl,(sekvec+1)
      ld    (unavec),a  ;unavec = sekvec
      ld    (unavec+1),hl
      ;
      ;check for write to unallocated sector
chkuna:     ld    a,(unacnt)  ;any unalloc remain?
      or    a
      jp    z,alloc           ;skip of not
      ;
      ;more unallocted records remain
```

```
      dec   a           ;unacnt=unacnt-1
      ld    (unacnt),a
      ld    a,(sekdsk)  ;same disk?
      ld    hl,unadsk
      cp    m           ;sekdsk=unadsk?
      jp    nz,alloc    ;skip if not
      ;
      ;disks are the same
      ld    a,(unavec)
      ld    hl,sekvec
      cp    m
      jr    nz,alloc    ;skip if not same hh(vec)
      ld    hl,(sekvec+1)
      ld    de,(unavec+1)
      or    a
      sbc   hl,de
      jr    nz,alloc    ;skip if not same h,l(vec)
      ;
      ;match, move to next sector for future ref
      ld    hl,1
      add   hl,de       ;unavec = unavec + 1
      adc   a,0
      ld    (unavec),a
      ld    (unavec+1),hl
      ;
      ;match found, mark as unnecessary read
noovf:      xor   a           ;0 to accumulator
      ld    (rsflag),a  ;rsflag=0
      jr    rwoper            ;to perform the write
      ;
      ;not an unallocated record, requires pre-read
alloc:      xor   a           ;0 to accum
      ld    (unacnt),a  ;unacnt=0
      inc   a           ;1 to accum
      ld    (rsflag),a  ;rsflag=1
      ;
      ;enter here to perform the read/write
rwoper:     xor   a           ;zero to accum
      ld    (erflag),a  ;no errors (yet)
      ld    hl,(sekvec+1)
      ld    a,bufmsk    ;buffer mask to A
      cpl               ;complement
      and   l           ;mask l byte of secvec
      ld    l,a
      ld    b,(ix+15)   ;load right shifts number
      ld    a,(sekvec)
      inc   b
      dec   b           ;shift number = 00 ?
      jr    z,savsekh   ;if yes
sphshft:srl a           ;shift sekvec SPH times right
      rr    h
      rr    l
      djnz  sphshft
savsekh:ld  (sekhst),a
      ld    (sekhst+1),hl     ;store host seek vector
      ;
      ;active host sector?
      ld    hl,hstact   ;host active flag
      ld    a,m
      ld    m,1         ;always becomes 1
      or    a           ;was it already?
      jr    z,filhst    ;fill host if not
      ;
      ;host buffer active, same aq seek buffer
```

```
      ld    a,(sekdsk)
      ld    hl,hstdsk   ;same disk?
      cp    m           ;sekdsk=hstdsk?
      jr    nz,nomatch
      ;
      ;same disk, same vec?
      ld    a,(sekhst)
      ld    hl,hstvec
      cp    m
      jr    nz,nomatch  ;skip if not
      ld    de,(sekhst+1)
      ld    hl,(hstvec+1)
      or    a
      sbc   hl,de
      jr    z,match           ;skip if match
      ;
      ;proper disk, but not correct vector
nomatch:ld  a,(hstwrt)  ;host written
      or    a
      call  nz,writehst ;clear host buff
      ;
      ;may have to fill the host buffer
filhst:     ld    a,(sekdsk)
      ld    (hstdsk),a
      ld    a,(sekhst)
      ld    hl,(sekhst+1)
      ld    (hstvec),a
      ld    (hstvec+1),hl     ;host seek to host address vec
      ld    a,(rsflag)  ;need to read?
      or    a
      call  nz,readhst  ;yes, if 1
      xor   a           ;0 to accum
      ld    (hstwrt),a  ;no pending write
      ;
      ;copy data to or from buffer
match:      ld    a,(sekvec+1)      ;l byte of seek vector
      and   bufmsk            ;buffer mask
      rra               ;same as *128
      ld    h,a
      ld    l,0
      rr    l
      ;
      ;HL has relative host buffer address
      ld    de,hstbuf
      add   hl,de       ;hl=host address
      ld    de,(dmaad)  ;get/put cp/m data
      ld    bc,128            ;length of move
      ld    a,(readop)  ;wich way?
      or    a
      jr    nz,rwmove   ;skip if read
      ;
      ;write operation, mark and switch direction
      ld    a,1
      ld    (hstwrt),a  ;hstwrt=1
      ex    de,hl       ;source /dest swap
      ;
rwmove:     ldir
      ;
      ;data has been moved to/from host buffer
      ld    a,(wrtype)  ;write type
      cp    wrdir       ;to directory?
      ld    a,(erflag)  ;in case of errors
      ret   nz          ;no further processing
      ;
```

```
      ;clear host buffer for directory write
      or    a           ;errors?
      ret   nz          ;skip if so
      xor   a           ;0 to accum
      ld    (hstwrt),a  ;buffer written
      call  writehst
      ld    a,(erflag)
      ret
      ;
      ;addressing the disk (in a) characteristics by IX reg.
dpbix:      ld    hl,dpbase+10-16   ;compute addr of according dph
      ld    de,16
dsearch:add hl,de
      sub   1
      jr    nc,dsearch
dfound:     ld    e,m
      inc   hl
      ld    d,m
      ld    ix,0
      add   ix,de       ;addr of actual dpb (a) to IX
      ret
;
      ;changes sektrk,seksec  to a log. vect.
logvec:     ld    a,(sekdsk)
      call  dpbix       ;address drive characteristics by IX
      ld    a,(ix+16)   ;are track's to corrigate ?
      or    a
      ld    hl,(sektrk) ;seek track to HL
      jr    z,nocorr    ;if not
      ld    e,(ix+13)   ;OFF to DE
      ld    d,(ix+14)
      sbc   hl,de       ;correction
nocorr:     ex    de,hl       ;corrected (or not) seek track's to DE
      ld    b,(ix+0)    ;cpm sectors per track
      xor   a
      ld    h,a
      ld    l,a
      ld    c,a         ;zero to 3 byte accumulator
bshft:      srl   b           ;how ls bit of 1st factor ?
      jr    nc,deshft   ;if low
      add   hl,de
      adc   a,c         ;3rd byte and carry add
deshft:     sla   e           ;prepare 2nd factor for next digit proc
      rl    d
      rl    c
      inc   b           ;1st factor 0 ?
      dec   b
      jr    nz,bshft    ;if not
      ld    de,(seksec) ;A=hh(logsec), H=h(logsec), L=l(logsec)
      ld    d,0         ;of previous tracks
      bit   0,(ix+18)   ;DEB = 1 ?
      jr    nz,nocorr1  ;no sector-1 computation for deblocked
      dec   de
nocorr1:add hl,de       ;add isolated sectors
      adc   a,0         ;involve carry
      ret
;
readhst:ld  a,(hstdsk)  ;READ to DISKIO
      call  dpbix
      ld    a,(ix+22)   ;get read com.
      jr    waitio
;
writehst:ld a,(hstdsk)  ;WRITE to DISKIO
      call  dpbix
```

```
      ld    a,(ix+22)   ;get read com.
      dec   a           ;make to write com.
;
waitio:
      ld    (cdb),a           ;store cdb byte   1 **********
;
phyvec:     xor   a           ;now compute TR a. SEC for BC
      ld    e,a
      ld    c,0ffh
      ld    d,(ix+31)   ;number of phy. sec/cyl. to D
      ld    hl,(hstvec+1)
      ld    b,8
comzyc:     rlc   c
      srl   d
      rr    e
      sbc   hl,de
      jr    nc,endtest
      res   0,c
      add   hl,de
endtest:djnz      comzyc
      ld    d,l
      ld    hl,cdb+2
      ld    m,c         ;store cdb byte 3 **********
      ld    a,(ix+16)   ;are track's to corrigate
      or    a
      jr    z,nocor           ;if not
      ld    a,(ix+13)   ;OFF
      add   a,c         ;correct the tracknumber
      ld    m,a
nocor:      inc   d           ;correct the sec.nr
      ld    a,e         ;phy.sec/track to a
      bit   0,(ix+21)   ;twosided disk ?
      jr    z,oneside   ;if not
      rra               ;phy.sec.nr / 2
      cp    d           ;haed 1 ?
      jr    nc,oneside  ;if not
      inc   b           ;b=1
      sub   d           ;compute sec.nr. for haed1
      neg
      ld    d,a
oneside:inc hl
      ld    m,b         ;store cdb byte 4 **********
      ld    c,b
      inc   hl
      ld    m,d         ;store cdb byte 5 **********
      ld    b,3         ;now store N,EOT,ALB in cdb
lodzyc:     inc   hl
      inc   ix
      ld    a,(ix+23)
      ld    m,a         ;store cdb byte 6,7,8 **********
      djnz  lodzyc
      ld    a,(hstdsk)
      call  dpbix
      ld    a,(ix+23)   ;LUN
      bit   0,c         ;haed1 active ?
      jr    z,haed0           ;no
      or    4           ;set haed1 in 2.byte CDB
haed0:      ld    (cdb+1),a   ;store cdb byte 2 **********
      ld    hl,cdb+8
      ld    a,(ix+17)   ;HPB
      ld    m,a         ;store cdb byte 9 **********
      ld    b,4
lodzyc1:inc hl          ;store PRAE,ALC,SRT,HLT
      inc   ix
```

```
      ld    a,(ix+26)
      ld    m,a         ;store cdb byte 10,11,12,13 *********
      djnz        lodzyc1
;
;
nready:     ld    a,(hstdsk)
      call  dpbix
      ld    l,(ix+19)   ;load dma address
      ld    h,(ix+20)
      ld    e,m
      inc   hl
      ld    d,m
      ld    hl,cdb
      call  diskio
      ld    a,b
      cp    'R'         ;drive ready ?
      jr    z,nready    ;if drive not ready
      or    a           ;any ERROR ?
      jr    z,erradr    ;if not
      ld    a,1
erradr:     ld    (erflag),a
      ret
;
      INCLUDE     DISKIO
;
idedc:      ds    32          ;EDC buffer
puff: db    0           ;check write buffer
si:   db    0,18h,2,8,1,0,4,44h,5,68h,3,0c1h
siok: db    5dh
siod: db      5ch
ctct:   db      58h
keyinit:ld  b,3           ;ignore keyboard-type after boot (wait 2 sec)
kwait:      ld    de,00h
kwait1:     dec   de
      ld    a,d
      or    e
      jr      nz,kwait1
      djnz  kwait
kini:   ;subroutine for outfinding of keyboard-ports and keyboard-initializing
      ld    a,(siok)
      ld    (keywait+1),a
      ld    (cint1+1),a     ;set keypoll-port in CTC-Interruptroutine
      ld    a,(siod)
      ld    (cint6+1),a     ;set keyin-port in CONIN
      ld    (keyfunc+2),a     ;set keycon-port in keyfunc
      ld    (keyled+4),a      ;set keycon-port in keyled
      ld    (keycin+1),a      ;set keyin-port in keywait
      ld    hl,ctct
      ld      c,m
      ld    a,5
      out     (c),a
      ld    a,1
      out   (c),a       ;ctc-initializing
      ld    hl,siok
      ld    c,m
      ld    b,0ch
      ld    hl,si
      otir              ;sio-initializing
      ld    hl,siod
      ld    c,m
      xor   a
        out       (c),a       ;reset to keyboard
      ld    b,3
px0:  ld    de,00h
```

```
      ld    hl,siok
px1:  ld    c,m
      in    a,(c)
      bit   0,a         ;character ready ?
      jr      nz,keyrd  ;if yes
      dec   de
      ld    a,d
      or    e
      jr      nz,px1
keyrd:      ld    hl,siod
      ld    c,m
      in    a,(c)       ;get character
      and   0f0h
      cp    80h         ;type ?
      jr    z,ctci0           ;if yes to ctc-init
      dec   b           ;repitition-counter -1
      jr    nz,px0          ;if not type and rep-count is not zero
      ld    hl,ctct
      ld    a,m
      cp    4ah
kerr: jr    z,kerr            ;keyboard is defect
      ld    m,59h       ;set new keyboard-ports in siok,siod,ctct
        cp  58h
      jr    z,var23
      ld    m,5ah
      cp    59h
      jr    nz,var4
var23:      dec   hl
      ld    m,52h
      dec   hl
      ld    m,53h
      jp    kini
var4: ld    m,4ah
      dec   hl
      ld    m,4eh
      dec   hl
      ld    m,4fh
      jp    kini

ctci0:      di                ;CTC-Initialisieren
      ld    a,0f8h            ;int-vector
      out   (0ch),a
      ld    a,0a5h            ;Steuerwort
      out   (0eh),a           ;Kanal 3
      ld    a,96
      out   (0eh),a           ;Zeitkonstante 10 ms
      ld    hl,cint0    ;Adr. der Interruptroutine
      ld    (0f7fch),hl
      ld    b,16
      ld    hl,keypuff
      xor   a
ctci1:      ld    (hl),a            ;keypuff loeschen
      inc   hl
      djnz  ctci1
      ei
      ret

;
      ;the remainder of the CBIOS is reserved uninitialized
      ;data area, and does not need to be a part of the system
      ;memory image (the space must be available, however,
      ;between "ecode" and "enddat").
ecode equ   $
      ds    1024-(ecode-puff)
```

```
;
hstbuf:     ds    1024        ;host buffer
;
sekdsk:     ds    1           ;seek disk number
sektrk:     ds    2           ;seek track number
seksec: ds  1           ;seek sector number
      ;
sekvec:     ds    3           ;log SMS vector get from sektrk & seksec
      ;
hstact: ds  1           ;host active flag
hstwrt: ds  1           ;host written flag
      ;
unacnt: ds  1           ;unallocated record count
unadsk: ds  1           ;last unallocated disk number
unavec:     ds    3           ;last unallocated records
      ;
hstdsk:     ds    1           ;host disk number
sekhst:     ds    3           ;log SMS vector for host to seek
hstvec:     ds    3           ;log SMS vector for read/write
      ;
erflag: ds  1           ;error flag
rsflag: ds  1           ;read sector flag
readop: ds  1           ;perform read operation if 1
wrtype: ds  1           ;write operation type
      ;
      ;scratch ram area for BDOS use
dirbf:  ds  128         ;scratch directory area
all00:      ds    76
all01:      ds    76
all02:      ds    76
all03:      ds    76
all04:      ds    76
chk00:      ds    64
chk01:      ds    64
chk02:      ds    64
chk03:      ds    64
chk04:      ds    64
      ;
keypuff:ds  16
enddat      equ   $           ;end of data area
endcode     equ   ecode       ;end of code
datsiz      equ   enddat-ecode      ;size of data area
      end
#############
```